INDUSTRIAL DATA 2002, (5) 1 : 49-54

DISEÑO DE UN INTÉRPRETE DE SQL

Hilmar Hinojosa L.*

RESUMEN

El presente artículo presenta en forma breve los tópicos más relevantes del diseño de un Intérprete del Lenguaje de Consultas Estructurales (SQL), el cual consiste en un programa que permite a un usuario crear y manipular una base de datos propia, haciendo uso de sentencias SQL estándar.

Palabras Clave: Lenguaje de Consultas Estructurales. Intérprete. Base de Datos

ABSTRACT

The present article presents in brief form the most excellent topics in the design of an Interpreter of SQL, which consists on a program that it allows an user to create and to manipulate an own database, making use of sentences standard SQL.

Key words : Structured Query Language. Interpreter. Database.

INTRODUCCIÓN

La explosiva popularidad de SQL es una de las tendencias más importantes en la industria informática hoy en día. Durante los últimos años SQL se ha convertido en el lenguaje de base de datos estándar. Todos los productos de gestión de base de datos soportan SQL, ejecutándose en sistemas informáticos que van desde computadoras personales a supercomputadoras. SQL ha saltado a un lugar prominente como tecnología informática importante y como fuerza de mercado poderosa.

SQL no es en sí mismo un sistema de gestión de base de datos ni un producto autónomo. No se puede ir a una tienda de informática y "comprar" SQL. En su lugar, SQL es parte integral de un sistema de gestión de base de datos, un lenguaje y una herramienta para comunicarse con el DBMS. Productos de este tipo son SQL Server, SQLBase, Oracle, Ingres, Sybase, etc. El costo de estos productos en el mercado es bastante elevado ya que básicamente están hechos para ser usados por empresas. Surge entonces el problema de que un usuario doméstico no pueda tener fácil acceso a los mismos. Bajo esta perspectiva el Intérprete de SQL que se propone constituye una posible opción, útil y económica.

*Magister en Ingeniería Industrial - Instituto de Investigación UNMSM
E-mail: iifi@unmsm.edu.pe

BASES DE DATOS RELACIONALES

Los sistemas de gestión de base de datos organizan y estructuran los datos de tal manera que puedan ser recuperados y manipulados por usuarios y programas de aplicación. Las estructuras de datos y las técnicas de acceso proporcionadas por un DBMS se denominan su modelo de datos. SQL es un lenguaje de base de datos para base de datos relacionales y utiliza el modelo de datos relacional.

Una base de datos relacional es una base de datos en donde todos los datos visibles al usuario están organizados estrictamente como tablas de valores y en donde todas las operaciones de la base de datos actúan sobre estas tablas.

SQL

SQL es una herramienta para organizar, gestionar y recuperar datos almacenados en una base de datos informática. El nombre SQL es una abreviatura de Structured Query Language (Lenguaje de Consultas Estructuradas). Específicamente SQL permite

interactuar con un tipo específico de base de datos llamada base de datos relacional. El programa informático que controla la base de datos se denomina Sistema de Gestión de Base de Datos (Database Management System) o DBMS.

Cuando se necesita recuperar datos de la base de datos, se utiliza el lenguaje SQL para efectuar la petición. El DBMS procesa la petición SQL, recupera los datos solicitados y los devuelve. Este proceso de solicitar datos de la base de datos y de recibir los resultados se denomina consulta (query), de aquí el nombre Structured Query Language.

A pesar de ello este nombre resulta inapropiado puesto que SQL es mucho más que una herramienta de consulta, aunque este fue su propósito original y una de sus funciones más importantes. SQL se utiliza para controlar todas las funciones que un DBMS proporciona a sus usuarios tales como definición, recuperación, manipulación, compartición e integridad de datos así como control de acceso

OPERACIÓN DE LAS SENTENCIAS DE SQL

A continuación se indican las principales operaciones que se implementan a través de sentencias SQL.

a. Creación de tablas

La sentencia CREATE TABLE permite definir una nueva tabla y la prepara para aceptar datos. Su sintaxis es la siguiente:

CREATE TABLE *nombre de tabla* (definición de columnas)

Las columnas de la tabla recién creada se definen en el cuerpo de la sentencia CREATE TABLE. Las definiciones de columnas aparecen en una lista separada por comas y encerrada entre paréntesis. El orden de las definiciones de las columnas determina el orden de izquierda a derecha de las columnas en la tabla. Cada definición especifica el nombre de la columna y el tipo de datos que la columna almacena.

b. Inserción de datos en una tabla

Típicamente, una nueva fila de datos se añade a una base de datos relacional cuando una nueva entidad representada por una fila aparece en el mundo exterior. Una fila es la unidad de datos más pequeña que puede añadirse a una tabla.

La sentencia INSERT permite añadir una nueva fila a una tabla. Su sintaxis es:

INSERT INTO *nombre tabla* (Lista de columnas)
VALUES (Lista de valores)

La cláusula INTO especifica la tabla que recibirá la nueva fila y las columnas que almacenarán los datos insertados. La cláusula VALUES especifica los valores de los datos que la nueva fila contendrá. Los nombres de columna y los valores de los datos se separan utilizando comas.

c. Modificación de datos de una tabla

Los valores de los datos almacenados en una base de datos se modifican cuando se producen cambios correspondientes en el mundo exterior, de tal forma que en todo momento la información almacenada en la base de datos sea un modelo exacto del mundo real. La unidad más pequeña que puede modificarse es una columna de una única fila.

La sentencia UPDATE permite modificar los valores de una o más columnas de las filas seleccionadas de una tabla. Debe respetarse la siguiente sintaxis:

UPDATE *nombre de tabla* SET *nombre de columna = expresión* WHERE *condición*

El nombre de tabla identifica la tabla en que se realizará la actualización. La condición que acompaña a la cláusula WHERE sirve para seleccionar las filas de la tabla que serán modificadas. La cláusula SET es una lista de asignaciones separadas por comas. Cada asignación identifica una columna destino a actualizar y especifica cómo calcular el nuevo valor para dicha columna.

d. Supresión de datos de una tabla

Normalmente, una fila de datos se suprime de una tabla cuando la entidad representada por la fila desaparece del mundo exterior. La unidad más pequeña de datos que puede ser suprimida es una única fila.

La sentencia DELETE permite eliminar filas seleccionadas de una sola tabla. La sintaxis de la sentencia DELETE es como sigue:

DELETE FROM *nombre de tabla* WHERE *condición*

La cláusula FROM especifica el nombre de la tabla de la que se eliminarán filas. La condición de la cláusula WHERE selecciona las filas que serán suprimidas. Obviamente si ninguna de las filas satisface dicha condición, ninguna de ellas será eliminada.

e. Consultas de datos

Las consultas de selección se utilizan para obtener información de las bases de datos, esta

información es devuelta en forma de conjunto de registros.

La sintaxis básica de una consulta de selección es la siguiente:

SELECT *Lista de Campos* FROM *Nombre de Tabla* WHERE *Condición*

En donde Lista de Campos es la relación de campos que se desean recuperar y Nombre de Tabla identifica a la tabla, origen de los mismos.

El resultado de una consulta SQL es siempre una tabla de datos, semejante a las tablas almacenadas en la base de datos. Generalmente los resultados de la consulta generarán una tabla con varias columnas y varias filas. Las consultas más sencillas solicitan columnas de datos de una única tabla en la base de datos.

f. Eliminacion de tablas

Cuando se quiere destruir una tabla existente que ya no es necesaria se debe utilizar la sentencia DROP TABLE cuya sintaxis es:

DROP TABLE *nombre de tabla*

El nombre de la tabla en la sentencia identifica la tabla a eliminar.

TRADUCTORES, COMPILADORES E INTÉRPRETES

Un traductor es cualquier programa que toma como entrada un texto escrito en un lenguaje, llamado fuente y da como salida otro texto en un lenguaje, denominado objeto.

En el caso de que el lenguaje fuente sea un lenguaje de programación de alto nivel y el objeto sea un lenguaje de bajo nivel (ensamblador o código de máquina), a dicho traductor se le denomina compilador.

Un intérprete no genera un programa equivalente, sino que toma una sentencia del programa fuente en un lenguaje de alto nivel y la traduce al código equivalente y al mismo tiempo lo ejecuta, es decir que se realiza una operación de compilación paso a paso. Luego se repite el mismo proceso con la sentencia siguiente.

ESPECIFICACIÓN DE UN INTÉRPRETE

La especificación de un intérprete se divide en tres partes:

a. Análisis Léxico

El analizador léxico, también conocido como scanner, lee los caracteres uno a uno desde la entrada y va formando grupos de caracteres con alguna relación entre sí (tokens), que constituirán la entrada para la siguiente etapa del compilador. Cada token representa una secuencia de caracteres que son tratados como una única entidad. Por ejemplo, en SQL un token es la palabra reservada SELECT.

Ya que es el que va leyendo los caracteres del programa, ignorará aquellos elementos innecesarios para la siguiente fase, como los tabuladores, comentarios, espacios en blanco, etc.

b. Análisis Sintáctico

El analizador sintáctico, también llamado parser, recibe como entrada los tokens que le pasa el Analizador Léxico (el analizador sintáctico no maneja directamente caracteres) y comprueba si esos tokens van llegando en el orden correcto (orden permitido por el lenguaje). Así pues, sus funciones son:

- Aceptar lo que es válido sintácticamente y rechazar lo que no lo es.
- Hacer explícito el orden jerárquico que tienen los operadores en el lenguaje de que se trate.
- Guiar el proceso de traducción (traducción dirigida por la sintaxis).

c. Análisis Semántico

El análisis semántico es posterior al sintáctico y mucho más difícil de formalizar que éste. Se trata de determinar el tipo de los resultados intermedios, comprobar que los argumentos que tiene un operador pertenecen al conjunto de los operadores posibles, y si son compatibles entre sí, etc. En definitiva, comprobará que el significado de lo que se va leyendo es válido.

GENERADOR DE ANALIZADORES SINTÁCTICOS YACC

Yacc (**Yet Another Compiler Compiler**) significa «otro compilador de compiladores más». Es un programa que permite construir analizadores sintácticos en lenguaje C a partir de una gramática libre al contexto.

Para construir un traductor utilizando YACC primero se prepara un archivo, por ejemplo fuente.c, que contiene una especificación en YACC del traductor.

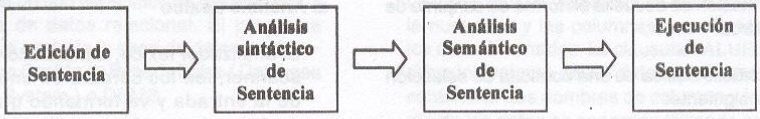


Figura 1. Secuencia de la sentencia.

yacc fuente.c: transforma al archivo fuente.c en un programa escrito en C llamado ytab.c, que implementa un analizador sintáctico escrito en C, junto con otras rutinas en C que el usuario pudo haber preparado en el archivo fuente.c. Posteriormente, se compila el archivo ytab.c y se obtiene el programa objeto deseado que realiza la traducción especificada por el programa original en YACC. Si se necesitan otros procedimientos, se pueden compilar o cargar con ytab.c, igual que en cualquier programa en C.

Un analizador gramatical construido en Yacc genera una función que realizará el análisis gramatical con el nombre de yyparse().

ALGORITMO IMPLEMENTADO PARA EL INTÉRPRETE DE SQL

Para poder realizar la interpretación de una sentencia SQL es necesario considerar las siguientes etapas: Ingreso de la sentencia haciendo uso de un editor, análisis sintáctico para comprobar la validez gramatical de la sentencia, análisis semántico para determinar el significado de la sentencia y la ejecución propiamente dicha de la sentencia.

El sistema consta de dos programas ejecutables: SQL.EXE y PARSQL.EXE. El programa SQL.EXE que ha sido escrito en Clipper provee al usuario el editor en el que es posible escribir la sentencia y ejecutarla.

El programa PARSQL.EXE es un programa compilado en lenguaje C que contiene el analizador sintáctico o parser para la gramática de SQL que se definió utilizando la herramienta YACC. Después de que se digita la sentencia SQL en el editor y se invoca su ejecución, se crea un archivo de texto llamado "SQL.TXT" que contiene el conjunto de caracteres que constituyen dicha sentencia. A continuación se llama al programa PARSQL.EXE el cual se encarga de procesar el archivo "SQL.TXT" verificando su validez gramatical.

Si no existe ningún error de tipo sintáctico se crea un archivo vacío denominado "SQL.OK". Cuando se regresa a la ejecución del programa SQL.EXE, éste realiza la comprobación de existencia del archivo "SQL.OK". Si existe, significa que la sentencia ha pasado satisfactoriamente el análisis sintáctico y se procede al análisis semántico.

De aquí en adelante el problema se reduce a manipular adecuadamente una cadena de caracteres para determinar el tipo de sentencia y reconocer sus elementos constitutivos para poder ejecutar su acción semántica.

En las figuras 2a y 2b se muestran las corridas del software.

CONCLUSIONES

Se ha desarrollado un sistema que ofrece al usuario la posibilidad de escribir sentencias de SQL y en caso de ser sintáctica y semánticamente correctas ejecuta las acciones correspondientes.

Se ha logrado plasmar en este producto el conocimiento de dos áreas fascinantes de la ciencia informática como son las Bases de Datos y la Teoría de Compiladores e Intérpretes.

Las consultas multitabla sólo pueden realizarse usando dos tablas como máximo. Como una mejora al trabajo desarrollado, se propone ampliar la capacidad para manejar mayor número de tablas.

El sistema organiza las tablas creadas haciendo uso de archivos tipo DBF, lo que lo hace compatible con tablas utilizadas en Fox y Clipper, permitiendo importar y exportar tablas desde y hacia dichos gestores de bases de datos. Queda abierta la posibilidad de hacer las interfaces necesarias para reconocer otros formatos de tablas.

El tiempo de respuesta de la ejecución de las sentencias es bastante breve.

El uso de YACC simplifica considerablemente la implementación del analizador sintáctico dada la complejidad de algunas sentencias, sobre todo la sentencia SELECT.

Se ha comprobado que la utilización de técnicas de programación adecuadas facilita la implementación de un proyecto complejo, dividiéndolo en módulos más simples.

El costo de desarrollo del software ha sido significativamente menor al que tiene cualquier producto similar que se encuentra en el mercado.

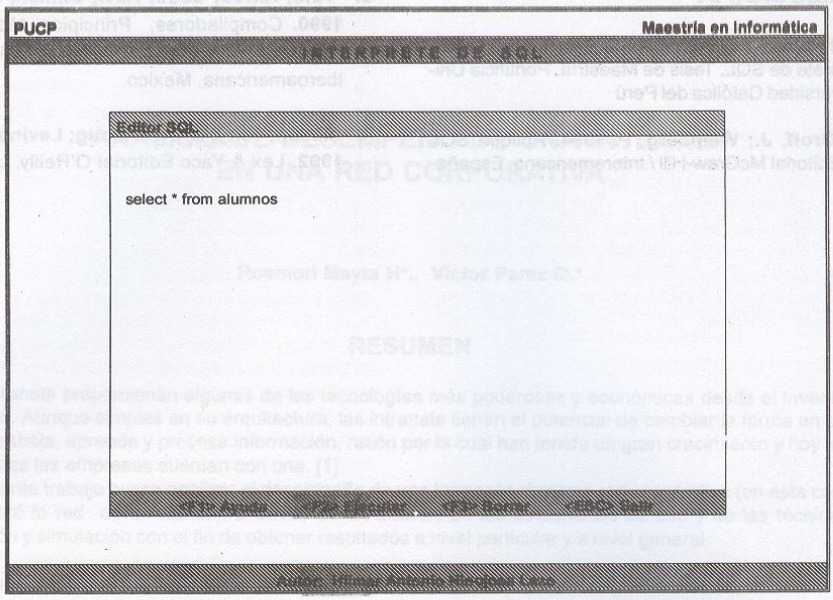
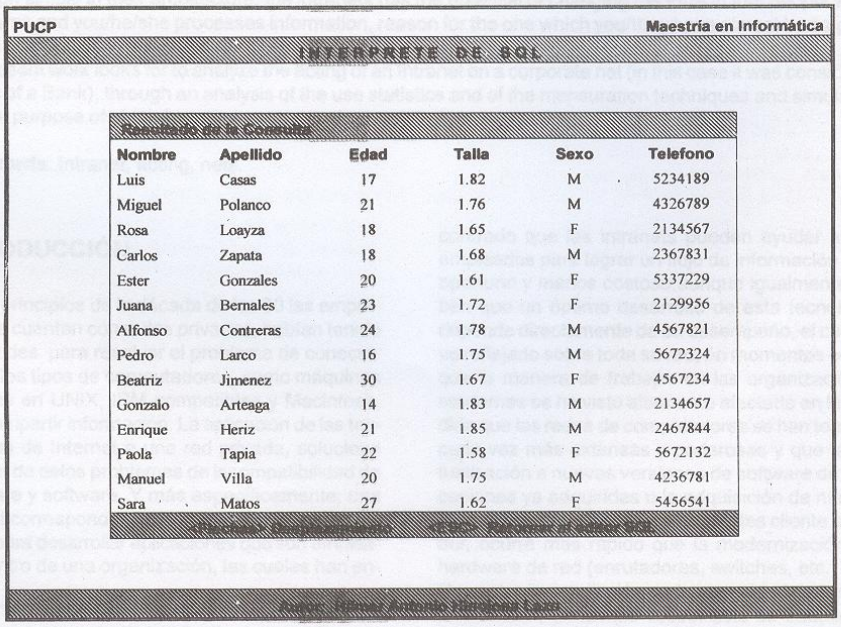


Figura 2a. Programa intérprete.

Después de pulsar la tecla F2 se realiza el análisis sintáctico y semántico de la sentencia, de no haber ningún error en breves instantes se muestra el resultado de la consulta



Nombre	Apellido	Edad	Talla	Sexo	Telefono
Luis	Casas	17	1.82	M	5234189
Miguel	Polanco	21	1.76	M	4326789
Rosa	Loayza	18	1.65	F	2134567
Carlos	Zapata	18	1.68	M	2367831
Ester	Gonzales	20	1.61	F	5337229
Juana	Bernales	23	1.72	F	2129956
Alfonso	Contreras	24	1.78	M	4567821
Pedro	Larco	16	1.75	M	5672324
Beatriz	Jimenez	30	1.67	F	4567234
Gonzalo	Arteaga	14	1.83	M	2134657
Enrique	Heriz	21	1.85	M	2467844
Paola	Tapia	22	1.58	F	5672132
Manuel	Villa	20	1.75	M	4236781
Sara	Matos	27	1.62	F	5456541

Figura 2b. Programa intérprete.

BIBLIOGRAFÍA

1. **Hinojosa L., Hilmar. 2002.** Diseño de un Intérprete de SQL. Tesis de Maestría. Pontificia Universidad Católica del Perú

2. **Groff, J.; Weinberg, P. 1994.** Aplique SQL Editorial McGraw-Hill / Interamericana. España.

3. **Aho, Alfred; Sethi, Ravi; Ullman, Jeffrey. 1990.** Compiladores, Principios, técnicas y herramientas Editorial Addison-Wesley Iberoamericana. México.

4. **Mason, Tony; Brown, Doug; Levine, John. 1992.** Lex & Yacc Editorial O'Reilly. U.S.A.